

Name of Book

Editors of Book

Graphical Models and Fusion in Sensor Networks

Müjdat Çetin Lei Chen John W. Fisher III Alexander T. Ihler O. Patrick Kreidl Randolph L. Moses Martin J. Wainwright Jason L. Williams Alan S. Willsky

MIT, The Ohio State University, and the University of California, Berkeley

0.1 Introduction

Graphical models provide a rich framework for capturing the statistical relationships among large numbers of variables, some of which may be measured and others of which are to be estimated or inferred from the available data. In addition, the natural algorithms for performing such inference tasks involve parallel message-passing procedures—essentially the passing of estimated statistical likelihoods—for the fusion of information across the entire graphical model. Because of both their graphical structure and the distributed nature of the inference procedures they admit, graphical models provide a natural framework to investigate fusion algorithms for sensor networks, an observation that has motivated a number of researchers, including the authors. Moreover, since the building or learning of such models, the analysis of such inference algorithms, and the development of enhanced algorithms is a rich, active, and growing field, there is a rich foundation on which to build methodologies for sensor network fusion algorithms.

The objective of this chapter is to confirm this observation and also to make clear that there are additional issues that arise in the context of sensor networks that require new questions to be asked that have not typically been part of the line of inquiry for graphical models.

Name of Book. Edited by A. Swami
© 2001 John Wiley & Sons, Ltd

This is a Book Title Name of the Author/Editor
© XXXX John Wiley & Sons, Ltd

The result is a new and very rich research area that has already added some important new results for sensor networks and, interestingly, also for graphical models.

In the next section of this chapter we present a concise introduction to some of the basic ideas underlying graphical models and their inference algorithms, as well as an important extension that generalized particle filtering to graphical models. Following this we introduce two sensor network applications that we use as vehicles to illustrate the methods. These applications are self-localization in sensor networks and distributed data association and object tracking. The discussion of these applications allows us to discuss and illustrate a concept of primary importance. Specifically, the mapping of a sensor network fusion problem to a graphical model is far from unique and different choices of graphical representations can have drastically different implications for the organization and effectiveness of the fusion algorithms that result.

The core of this chapter is the description of research that deals with several of the key, new issues that arise in adapting and extending graphical model inference algorithms to sensor networks. In particular, we describe two approaches—message censoring and efficient communication of particle-based messages—aimed at addressing the fact that communication resources in sensor networks are generally severely limited. Such methods introduce errors into the transmitted messages (in order to conserve communication resources), and we summarize new results analyzing the impact of such errors on overall fusion performance, providing a complete audit trail from bits to fusion accuracy. We then examine a second very important issue, namely the fact that there is considerable flexibility in how inference computations are distributed among sensor nodes, leading (especially in tracking applications) to problems of the handoff of inference responsibilities. We do this in the context of power-constrained resource allocation, in which we must account not only for the power consumed in taking and communicating messages but also the power required for sensor handoff. All of the methods described so far involve the approximation of standard message-passing algorithms to accommodate power constraints. The last line of research we describe is that of completely redesigning message passing by taking into account from the start that there are bit-limitations on transmissions. This work, which makes contact with the field of decentralized team theory, also makes clear that there is a communication cost if sensors must organize themselves to achieve joint objectives. Our presentation of these methods is necessarily concise, and we provide ample references to complete treatments of each of these lines of investigation. We close the chapter with a brief discussion of some research directions that we believe are critical for the way forward.

0.2 Graphical Models

We begin with a brief discussion of graphical models, focusing on aspects related to inference and linking these to distributed inference in sensor networks in later sections. In this chapter we focus exclusively on the class of graphical models defined on undirected graphs (i.e., in which there is no parent-child relationship between the nodes connected by any edges). Such models are also commonly referred to as Markov random fields. There is also a very important class of graphical models defined on directed graphs – sometimes referred to as Bayesian networks – and we refer the reader to the literature (e.g., the book by Pearl Pearl (1988)) for development for these models. Also, our discussion here is, perforce, brief and

focused on the characteristics and results we require in our development. We refer the reader to the references, especially to the survey article and books Jordan (1998, 2004, in preparation); Lauritzen (1996); Whittaker (1990), for complete developments of this very important area.

0.2.1 Definitions and Properties

A *graphical model* is specified by an undirected graph, $G = (V, E)$, consisting of a vertex or node set V , and a set of edges $E \subset V \times V$. Associated with each node $v \in V$ is a random variable X_v ; the full collection $X = \{X_v, v \in V\}$ of random variables must collectively satisfy a set of Markov properties with respect to G . Specifically, for any subset U of V , let $X_U = \{X_v, v \in U\}$. We say that the random vector X is Markov with respect to G if for any partition of V into disjoint sets A, B, C , in which B *separates* A and C (i.e., all paths in G from A to C include vertices in B), the random vectors X_A and X_C are conditionally independent given X_B . For the “graph” associated with time series—i.e., consecutive points in time with each point connected to its immediate predecessor and successor—this corresponds to the usual notion of temporal Markovianity (i.e., that the past and future are conditionally independent given the present). For general graphs, however, the Markov property requires a far richer set of conditional independencies and associated challenges in both specifying such distributions and in performing inference using them. By way of example, consider the graph of Figure 7(a) (used for subsequent analysis) in which the variables x and y are conditionally independent given variables w and z . However, they w and z are *not* conditionally independent given x and y due to the edge between w and z .

The celebrated Hammersley-Clifford Theorem (Brémaud 1991) provides a sufficient condition (also necessary for strictly positive probability distributions) for the form that the joint distribution must take in order to be Markov with respect to G . Specifically, let \mathcal{C} denote the set of all *cliques* in G , where a subset of nodes C is a clique if it is fully connected (i.e., an edge exists between each pair of nodes in C). The random vector X is Markov with respect to G if (and only if for strictly positive probability distributions) its distribution admits a factorization as a product of functions of variables restricted to cliques of the form

$$p(x) = \frac{\prod_{C \in \mathcal{C}} \psi_C(x_C)}{Z}, \quad \text{where} \quad Z \triangleq \sum_x \prod_{C \in \mathcal{C}} \psi_C(x_C) \quad (1)$$

is the *partition function*, and the $\psi_C(x_C)$ are so-called *compatibility functions*. The logarithms of these compatibility functions are commonly referred to as *potentials* or *potential functions*.

For simplicity we will assume for the remainder of this paper that each of the nonzero potentials (or equivalently each compatibility function in (1) that is not constant) is a function either of the variable at a single node of the graph (*node potentials*) or of the variables at a pair of nodes corresponding to an edge in E (*edge potentials*). In this case, (1) takes the form:

$$p(x) = \frac{\left(\prod_{s \in V} \psi_s(x_s)\right) \left(\prod_{(s,t) \in E} \psi_{s,t}(x_s, x_t)\right)}{Z} \quad (2)$$

Note that any graphical model can be put into this form by appropriate node aggregation (Brémaud 1991). While all of the ideas that are presented here can be extended to the more general

case, pairwise potentials are sufficient for the specific applications considered in this chapter. Moreover, the communication interpretation of the so-called message-passing algorithms used herein are more easily explained in this context.

As long as G is a relatively sparse graph, the factorizations (1) or (2) represent parsimonious means to describe the joint distribution of a large number of random variables—in the same way that specifying an initial (or final) distribution and a set of one-step transition distributions is a compact way in which to specify a Markov chain. Moreover, for many inference and estimation problems (including those described in this chapter), such a specification is readily available. The challenge, however, is that unless the graph has very special properties—such as in the case of Markov chains—the compatibility functions do not readily describe the quantities of most interest, such as the marginal distribution of the variables at individual (or small sets of) nodes or the overall peak of the distribution jointly optimized over all nodes. Indeed, for discrete-valued random variables the computation of such quantities for general graphs is NP-Hard.

0.2.2 Sum–Product Algorithms

For graphs without cycles (Markov chains and, more generally, graphical models on trees), computation of the marginal distributions is relatively straightforward. In this case, the node and pair-wise potentials of the joint distribution in (2) for any cycle-free graph can be expressed in terms of the marginal probabilities at individual nodes and joint probabilities of pairs of nodes connected by edges (Cowell et al. 1999; Wainwright et al. 2003):

$$p(x) = \prod_{s \in V} p_s(x_s) \prod_{(s,t) \in E} \frac{p_{st}(x_s, x_t)}{p_s(x_s)p_t(x_t)}, \quad (3)$$

That is, $\psi_s(x_s) = p_s(x_s)$ (or $\psi_s(x_s) = p_s(x_s)p(y_s|x_s)$ when there is a measurement y_s associated with x_s) and $\psi(x_s, x_t) = \frac{p(x_s, x_t)}{p(x_s)p(x_t)}$. Marginal probabilities can be efficiently calculated in a *distributed* fashion by so-called sum–product algorithms. Specifically, as shown in (Pearl 1988), the marginal probabilities at any node s in the graph can be expressed in terms of the local potential ψ_s at node s , along with a set of so-called *messages* from each of its neighbors in the set $\mathcal{N}(s) = \{t \in V \mid (s, t) \in E\}$. The message from node t to node s is a function $M_{ts}(x_s)$ that (up to normalization) represents the likelihood function of x_s based on the subtree rooted at t and extending away from s . In particular, the marginal distribution p_s takes the form

$$p_s(x_s) \propto \psi_s(x_s) \prod_{t \in \mathcal{N}(s)} M_{ts}(x_s). \quad (4)$$

Furthermore, in the absence of cycles, these messages are related to each other via a sum–product formula:

$$M_{ts}(x_s) \propto \sum_{x_t} \psi_{st}(x_s, x_t) \psi_t(x_t) \prod_{u \in \mathcal{N}(t) \setminus s} M_{ut}(x_t). \quad (5)$$

The product operation embedded in the message computation from node t to s combines the information in the subtree rooted at node t , combining the likelihood information from all neighbors of node t other than s with the local potential at node t . This yields a likelihood

function for the random variable X_t at node t . This is then converted to a likelihood for the random variable X_s at node s by multiplying by the compatibility function between these two nodes and then “summing” or integrating out the variable at node t in a fashion analogous to the Chapman-Kolmogorov equation in a Markov chain.

Together equations (4) and (5) relating messages throughout the loop-free graph represent a set of fixed-point equations that can be solved in a variety of ways corresponding to different message-passing algorithms. For example, one can solve these equations explicitly, much as in Gaussian elimination, by starting at leaf nodes, working inward toward a “root” node, and then propagating back toward the leaves—this is a generalization of two-pass smoothing algorithms for Markov chains. An alternative is to solve these equations iteratively: we begin with guesses (often taken simply to be constant) of all of the messages and iteratively update messages by substitution into the fixed-point equations. Each step of this procedure involves passing the current guess of messages among neighboring nodes. While there is great flexibility in how one schedules these messages, the happy fact remains that after a sufficient number of iterations (enough so that information propagates from every node to every other), the correct messages are obtained from which the desired probabilities can then be computed.

0.2.3 Max-Product Algorithms

Interestingly, for loop-free graphs, a variant of this approach also yields the solution to the problem of computing the overall *maximum a posteriori* (MAP) configuration for the entire graphical model. For such graphical models, there is an alternative factorization of $p(x)$ in terms of so-called *max-marginals*. As their name would suggest, these quantities are defined by eliminating variables through maximization (as opposed to summation); in particular, we define

$$q_s(x_s) := \max_{x_u, u \in V \setminus s} p(x_1, \dots, x_n) \quad (6a)$$

$$q_{st}(x_s, x_t) := \max_{x_u, u \in V \setminus \{s, t\}} p(x_1, \dots, x_n). \quad (6b)$$

It is a remarkable fact that for a tree-structured graph, the distribution (1) can also be factorized in terms of these max-marginals—viz.:

$$p(x) \propto \prod_{s \in V} q_s(x_s) \prod_{(s, t) \in E} \frac{q_{st}(x_s, x_t)}{q_s(x_s)q_t(x_t)}. \quad (7)$$

Furthermore, there are equations analogous to those for the sum-product algorithm that show how these quantities can be computed in terms of node potentials and messages, where the fixed point equations involve maximization rather than summation (yielding what are known as *max-product* algorithms). The solution of these fixed point equations can be computed via leaf-root-leaf message passing (corresponding to dynamic programming/Viterbi algorithms Forney (1973)) or by iterative message passing with more general message scheduling.

0.2.4 Loopy Belief Propagation

While the representation of marginal distributions or max-marginals at individual nodes in terms of messages holds only if the graph is cycle-free, equations (4) and (5) are well-defined

for any graph, and one can consider applying the sum-product algorithm to arbitrary graphs which contain cycles (often referred to as *loopy belief propagation*), this corresponds to fusing information based on assumptions that are not precisely valid. For example, the product operation in equations (4) as well as (5) corresponds to the fusion of information from the different neighbors of a node, t , assuming that the information contained in the messages from these different neighbors are conditionally independent given the value of x_t , something that is valid for trees but is decidedly *not* true if there are cycles.

Despite this evident suboptimality, this loopy form of the sum-product algorithm has been extremely successful in certain applications, most notably in decoding of low-density parity check codes (Kschischang 2003; Richardson and Urbanke 2001) which can be described by graphs with long cycles. In contrast, many sensor network applications (as well as others) involve graphs with relatively short cycles. This has led to a considerable and still growing body of literature on the analysis of these algorithms on arbitrary graphs, as well as the development of new ones that yield superior performance. For arbitrary loopy graphs, the reparameterization perspective on these algorithms (Wainwright et al. 2003), in conjunction with a new class of efficiently computable bounds (Wainwright et al. 2005b) on the partition function Z , provide computable bounds on the error incurred via application of sum-product to loopy graphs. Similar analysis is also applicable to the max-product updates (Freeman and Weiss 2001; Wainwright et al. 2004). The fact that the max-product algorithm may yield incorrect (i.e., non-MAP) configurations motivates the development of a new class of *tree-reweighted max-product algorithms* (TRMP) (Wainwright et al. 2005a) for which—in sharp contrast with the ordinary max-product updates—there are a set of testable conditions for determining if the solution is indeed the MAP configuration. Tight performance guarantees can be provided for TRMP for specific classes of graphs (Feldman et al. 2005; Kolmogorov and Wainwright 2005). More broadly, we refer the reader to various research and survey papers, e.g. (Loeliger 2004; Wainwright and Jordan 2005; Yedidia et al. 2005), as well as citations at the end of this chapter, which provide only a sampling of this rapidly growing literature.

0.2.5 Nonparametric Belief Propagation

We close with a brief discussion of an algorithm which will play an important role in later sections. Whether applied to loop-free or loopy graphs, message-passing algorithms corresponding to the iterative solution of (5) require the transmission of a full likelihood function, parameterized by the variable at the receiving node. When those variables are discrete (take on finitely many values), the message can be represented as a vector of numbers; when the variables are jointly Gaussian, they can be described by means and covariances. However, for non-Gaussian continuous variables, in principle we need to transmit an entire continuous function. One common approach is to discretize the underlying continuous variables. Unfortunately, this can often lead to high (and unwarranted) computational complexity (and in sensor networks, a high communication overhead as well). In nonlinear, non-Gaussian systems defined on Markov chains, particle filtering is often used to avoid these high computational and representation costs. Here, we describe a recently developed generalization of particle filtering to more complex graphical models, called *Nonparametric Belief Propagation*, or NBP (Sudderth et al. 2003).

Particle filtering also involves an equation similar to (5), with one very important simplification: there is no “product”, since there is only one “other” neighbor of node t . The algorithm takes a set of particles, representing samples from the single message corresponding to the product term in (5), weights these particles by the local node’s compatibility function (and perhaps resamples from this weighted collection of particles), and then performs the “sum” operation by simulating the transition dynamics from node t to s , i.e., for each sample at node t we sample from the transition distribution to generate a sample at node s .

All of these steps, with one significant exception, apply equally well to message passing in general graphical models. The difficulty, however, arises from the fact that we have the product of *several* particle-based messages (one from each neighbor) at each node. As developed in (Sudderth et al. 2003), in order to make sense of this product one can interpret each particle-based message as a nonparametric estimate of the likelihood function/probability density corresponding to the exact message, which can be smoothed to create a well-defined product. For example, if we use Gaussian kernels for these nonparametric densities, the set of particles corresponds to a Gaussian sum. As a result, the problem of generating samples from the product of messages, each represented by a set of particles, reduces to that of drawing samples from a density defined by the product of a collection of Gaussian sums.

This operation presents some computational difficulty: unless we are careful, we may have a geometrically growing number of terms in the aforementioned Gaussian sums, too many to work with efficiently. The trick to drawing samples efficiently is to draw them without ever explicitly constructing the true product. Specifically, one can generate a sample in two steps: first, choose one of the Gaussian mixture components in the product from which to sample (i.e., the set of labels corresponding to one component in each of the incoming messages), and then drawing a sample from the Gaussian corresponding to that set of labels. Since this latter step is straightforward, the challenge is in the former (sampling the labels). As one solution, importance and Gibbs sampling methods can be applied to draw a label without enumerating all possible combinations.

Additionally, the sampling process can be made dramatically more efficient by using a multi-resolution representation based on the so-called k -dimensional (or KD-) trees (a structure that will also play a key role in efficient coding and communication of messages in Section 0.4.2). Given a set of k -dimensional particles (i.e., x_s is a k -dimensional real-valued random variable) representing one of the messages in (5), we create a multi-resolution hierarchical clustering of the particles, beginning with all particles clustered together at the root of the tree. Proceeding downward through the tree, at each node we subdivide that node’s cluster into two sub-clusters (associated with the children of the current node) by splitting the data along one of the k dimensions (and cycle through each of these dimensions as we proceed down the tree) until at the finest level of the tree we have individual particles. The sampling operations described previously can then be accomplished very efficiently using this coarse-to-fine representation; we refer the reader to (Ihler et al. 2003) for details.

0.3 From Sensor Network Fusion to Graphical Models

In this section we describe how one maps a fusion problem involving a network of sensors into an inference problem on a graphical model. Such a mapping might at first seem trivial, as

a natural graph already exists, defined by the sensor nodes and the inter-sensor communication structure. However, it is the *informational structure* of the inference problem—involving the relationships between sensed information and the variables about which we wish to perform estimation—that forms the basis of the mapping, and is just as critical as the communication structure of the problem. We illustrate this mapping by describing two sensor networks applications, focusing primarily on their informational structure.

0.3.1 Self-Localization in Sensor Networks

A crucial first step for many sensor network applications is that of sensor localization—determining the locations of sensor nodes in a network. Figure 1 illustrates the localization problem. Each node corresponds to a sensor, and the random vector at that node describes sensor location, and also perhaps other calibration variables such as the sensor’s orientation (e.g., if the node provides directional-sensing capability), and the time offset of its internal clock (e.g., if inter-sensor time-of-flight measurements are used by the network).

To simplify the discussion we will consider only location variables, although the framework we describe extends immediately to a more general setting. We consider a case in which the available information for estimating sensor locations consists of: (i) uncertain prior information about the location of a subset of the sensors (e.g., if any of the sensors are provided with GPS); (ii) how likely sensors are to ‘hear’ one another and attempt to measure their inter-sensor distance (typically only possible for nearby pairs of sensors); and (iii) any distance measurements so obtained. Specifically, let us denote by $\rho_s(x_s)$ the prior location probability distribution for sensor s , if any, let $Pr(x_s, x_t)$ be the probability of observing a distance measurement between two sensors s, t located at x_s and x_t , and let $\rho_L(l_{st}|x_s, x_t)$ be the probability distribution the measured distance l_{st} given that the true sensor positions are x_s and x_t . Notice that all three sources of information involve only the variables at a single sensor or at pairs of sensors; thus we may use a pairwise graphical model to describe the joint distribution of sensor locations, with each node in the graph associated with one sensor and its location or calibration variables of interest. One may thus immediately write that the joint probability distribution has the form of (2), with

$$\psi_s(x_s) = \rho_s(x_s) \tag{8a}$$

$$\psi_{st}(x_s, x_t) = \begin{cases} Pr(x_s, x_t)\rho_L(l_{st}|x_s, x_t) & ; \text{if } l_{st} \text{ is observed} \\ 1 - Pr(x_s, x_t) & ; \text{otherwise} \end{cases} \tag{8b}$$

If we momentarily ignore the information content of aspect (ii) (as is common in many localization problems, though we shall discuss its inclusion subsequently) we immediately obtain the graphical model shown in Figure 1(b), in which an edge connects each pair of sensors which are able to measure their distance. In this case, single-node potentials are included for those sensors with prior location information, and edge potentials represent the likelihood function for the locations of the two sensors based on the observed distance between them. Note that the above graphical model structure depends on the informational structure of the measurements, rather than on the communication structure of the network.

The sensor localization problem is then precisely one of computing the best estimates of all sensor locations given the prior and measured information. As an optimization problem this has been well-studied by others, (Moses et al. 2003; Patwari et al. 2003; Thrun 2006),

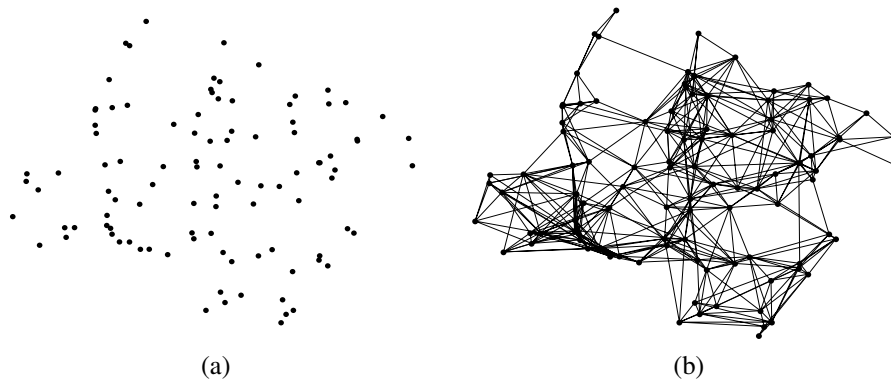


Figure 1 Sensor localization: (a) the physical location of a collection of sensors may be represented as (b) a graphical model in which each node corresponds to a sensor's location variables and each edge corresponds to observed information, such as the inter-sensor distance measurements between a pair of sensors. (Reproduced by permission of ©2006 IEEE)

often under the assumption that the distributions (ρ_s, ρ_L) involved are Gaussian so that its solution entails the nonlinear optimization of a quadratic cost function to obtain a point estimate. However, by formulating this problem as one of inference for a graphical model (in which localization corresponds to computing the marginal distributions of variables at each node), we directly obtain message-passing algorithms (such as sum-product) that distribute the computations across the network. The resulting solution is an estimate of the marginal distribution of the location variables, rather than a point estimate or confidence interval, and thereby provides a detailed statistical description of the localization solution.

Moreover, the graphical model formulation allows us to easily include features which bring additional realism, accuracy, and well-posedness to the problem (Ihler et al. 2005b; Ihler 2005). In particular, anomalous range measurements can occur with nontrivial probability; for example, anomalies can occur when the direct path between sensors is obscured, or when malicious signals are sent to thwart a localization process. In the graphical model framework, anomalous estimates are easily handled through a simple modification of the edge potential likelihood functions to capture such measurement errors. Also, location distributions for sensors can be multi-modal even when measurements are Gaussian—for example, receiving perfect range measurements from two neighboring sensors whose locations are known perfectly yields two possible locations. The NBP formulation finds an estimate of the node location probability distributions—as opposed to a point estimate such as the distribution maximum—from which multi-modal, non-Gaussian, or other characteristics are readily seen. Representing such multi-modality at the individual node level is one of the strengths of the NBP algorithm, and is far easier than dealing with this at a centralized level.

The graphical model formulation of the localization problem also provides a natural mechanism for including uncertain information about *which* pairs of sensors are able to measure distance. This information can improve the well-posedness and reduce estimation error in the location estimates. For example, the topmost sensor node in Figure 1 forms a location

estimate using distance measurements from two very closely-located sensors. Thus, the probability distribution for the location estimate will have large values in an entire circular region centered at the midpoint of these two other sensors. However, if we also account for the fact that the node on the other side of these two sensors *cannot* hear (or can hear only with small probability) the topmost sensor, this circular ambiguity is considerably reduced. Incorporating the absence of a measurement as an indicator of sensors being more distant from each other can be difficult to accommodate in the traditional optimization framework (for example, it is certainly not a simple quadratic cost). In this graphical model formulation, however, this information can be included simply by adding edges to the graph. Specifically, we include 2-step (or more generally, n -step) edges, where a 2-step edge is one between sensors that are heard by a common sensor but cannot hear each other (Ihler et al. 2005b). Including these additional edges increases the complexity of inference slightly, although experimentally it appears that a few edges are often sufficient to resolve most of the ambiguities which may arise.

The fact that the random variables in the localization graphical model are continuous-valued—leading to our use of the particle-based NBP algorithm—raises a number of questions unique to sensor networks related to how best to use scarce communication resources. For example, how many particles do we need to send, and how can we encode them efficiently? Moreover, how do we adapt these choices as estimates evolve dynamically (e.g., as ambiguities due to multi-modality in distributions resolve themselves)? These are among the issues considered in Section 0.4.

0.3.2 Multi-Object Data Association in Sensor Networks

A second application for sensor networks is that of multi-sensor, multi-object tracking. This is a challenging problem even for centralized algorithms in large part because of the embedded problem of data association, i.e., of determining which measurements from different sensors correspond to the same object. For sensor networks there are additional challenges, due to the need for distributed implementation, but typical networks also have structure; e.g. sensors have limited sensing range overlapping the range of a limited number of other sensors. This suggests new approaches for solving data association problems that are computationally feasible and fully distributed.

Figure 2 depicts a notional example of the problem. Here 25 sensors cover a region of interest with overlapping areas of regard. Several targets are located within the region, each sensed by one or more sensors. In this case, the mapping of the inference problem to a graphical model is not unique. Different mappings reflect computational tradeoffs leading to different solutions than in typical centralized multi-target tracking approaches. In particular, a widely used centralized approach (Kurien 1990) organizes data association hypotheses based on so-called track hypotheses, leading to data structures—and corresponding graphical models—in which, roughly speaking, the nodes correspond to targets.

In a sensor network, however, it is advantageous to organize the representation around sensors rather than targets. For centralized processing, such measurement-oriented approaches have been discarded for the same basic reason that purely sensor-based representations do not work here. In particular, consider a simple situation in which we know how many targets are present and we know which sensors see which targets. If we wish to use a model in which the nodes are in 1-1 correspondence with the sensors, the variable to be estimated at each node is

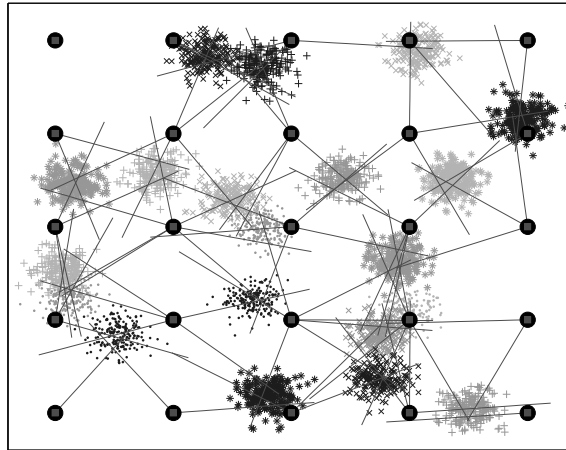


Figure 2 A snapshot of a typical data association scenario in a sensor network. 25 sensors (circle nodes) and the bearing-only measurements (line segments) are shown. Prior distributions of target locations are represented nonparametrically with samples from the individual marginal target distributions (appearing as clusters). (Reproduced by permission of ©2006 IEEE)

simply the association vector that describes which measurement from that sensor goes with which target, which measurements are false alarms, and which targets in its area of regard it fails to detect. The problem with such a graphical model is that if multiple targets are seen by the same set of sensors, the likelihoods of these sets of associations (across sensors and targets) are coupled, implying that in the representation in (1), we must include cliques of size larger than two. In principle, these cliques can be quite large, and it is precisely for this reason that the association problem is NP-Hard.

By taking advantage of the sparse structure of sensor networks; i.e., the fact that each sensor has only a limited field of view and thus has only a modest number of other sensors with which it interacts and small number of targets within its measurement range—one can readily construct a hybrid representation comprised of two types of nodes. *Sensor nodes* capture the assignment of groups of measurements to *multi-target nodes* in addition to assignments that do not have any multi-sensor/target contention. *Multi-target nodes* corresponding to sets of targets seen by the same set of 3 or more sensors. In such a model, the variable at each sensor node captures the assignment of groups of measurements to each of these multi-target nodes as well as any assignments that do not have such a level of multi-sensor/target contention. Moreover, the resulting graphical model yields a representation as in (2) with only pairwise potentials (Chen et al. 2005b). Furthermore, while we have described the idea for the case in which we already know which targets are seen by which sets of sensors, it is also possible to formulate graphical models that deal with the problem of also determining which targets are seen by which subsets of sensors. We do so by introducing virtual nodes representing regions of space corresponding to overlaps in areas of regard of multiple sensors, as shown in Figure 3. In addition, although we have discussed the data association problem at a single

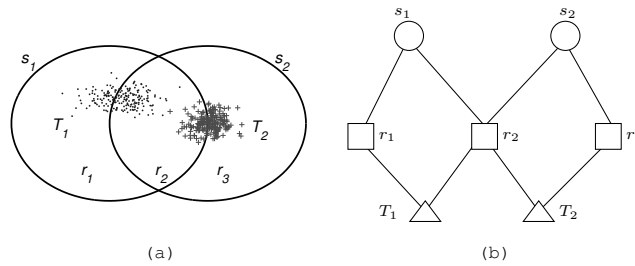


Figure 3 (a) A piece of a partially organized sensor network, where the sensor-target coverage relationship is ambiguous. Two sensors with their surveillance regions (s_1 and s_2). Prior distributions of *two* targets are represented nonparametrically as samples from and individual marginal target distributions (T_1 and T_2). The surveillance area is divided into three non-overlapping subregions (r_1 – r_3), each of which is covered by a distinct subset of sensors. (b) The graphical model for the scenario in (a); circles, squares, and triangles correspond to sensors, subregions, and targets, respectively. (Reproduced by permission of ©2006 IEEE)

time point here for simplicity, the tracking problem is of course dynamic, and our framework can be generalized to incorporate data from multiple time slices by using a multiple hypothesis tracking-like approach (Chen et al. 2005a).

For the data association problem, both the computation of marginal probabilities and of the overall MAP estimate are of interest. The MAP estimate is of importance because it captures consistency of association across multiple sensors and targets (for example, capturing the fact that one measurement cannot correspond to multiple targets). As a result, algorithms such as sum-product and max-product are both of interest, as is the issue of communications-sensitive message passing, a topic to which we turn in the next section.

0.4 Message Censoring, Approximation, and Impact on Fusion

The sensor network applications in the previous section are two of many that can be naturally cast as problems of inference in graphical models—at least in part, as there are other issues, including power conservation and careful use of scarce communication resources, that must be considered. Using the two previous applications as examples, we describe approaches to dealing with such power and communication issues.

0.4.1 Message Censoring

As described in the preceding section, multi-object data association in sensor networks can be formulated as a problem either of computing the marginal probabilities or the overall MAP estimate for a graphical model whose variables are discrete and represent various assignments of measurements to objects or spatial regions. In our work we have applied

a variety of different algorithms to solve this problem, including the sum–product algorithm (Kschischang et al. 2001) for the computation of approximate marginals, the max–product algorithm (Wainwright et al. 2004) for the computation of approximate MAP estimates and the TRMP algorithm (Wainwright et al. 2004) for the computation of the true MAP estimate.

One issue with all of these algorithms is that messages are, in principle, continually transmitted among all of the nodes in the graphical model. In typical graphical model applications some type of global stopping rule is applied to decide when the inference iterations should be terminated. Also, it is often the case that convergence behavior can depend strongly on the *message schedule*—i.e., the order in which messages are created, transmitted, and processed. For sensor network applications, convergence criteria or message schedules that require centralized coordination are out of the question. Rather, what is needed are local rules by which individual nodes can decide, at each iteration, whether it has sufficient new information to warrant the transmission of a new message, with the understanding that the receiving node will simply use the preceding message if it does not get a new one, and will use a default value corresponding to a noninformative message if it has not received any prior message. This formulation also allows for transmission erasures.

A simple local rule for message censoring is the following: first, we interpret each message as a probability distribution on the state of the node to which the message is to be sent; easily accomplished by normalizing the message. We then compute the Kullback-Leibler Divergence (KLD) between each message and its successor,

$$D(M_{ts}^k \| M_{ts}^{k-1}) = \sum_{x_s} M_{ts}^k(x_s) \log \frac{M_{ts}^k(x_s)}{M_{ts}^{k-1}(x_s)}, \quad (9)$$

as a measure of novel information and send M_{ts}^k only if $D(M_{ts}^k \| M_{ts}^{k-1})$ exceeds a threshold ϵ . This procedure is completely local to each node and provides for network adaptivity, as these rules lead to data-dependent message scheduling; indeed, it is quite common for a node to become silent for one or more iterations and then to restart sending messages as sufficiently new information reaches it from elsewhere in the network.

When applying the above method to algorithms such as sum–product, we observe that major savings in communication (hence power) can be achieved with modest performance loss as compared to standard message passing algorithms. An example is shown in Figure 4, where the data are obtained by simulating tracking of 50 targets in a 25 sensor network and censored versions of max–product are readily compared to max-product and TRMP. In the figure, both communication cost and data association error are shown for censored versions of the max–product algorithm as the censoring threshold is varied. Also shown are the communication and association error for the standard max–product and TRMP algorithms. It can be seen that for certain thresholds the data association performance loss is very small, while the amount of communication is dramatically reduced. This shows that censored message passing can provide significant communication savings together with near-optimal performance. In addition, we have found examples in which this algorithm yields *better* performance than one without message censoring. We conjecture this is related to the so-called “rumor propagation” behavior of algorithms such as sum–product in which the repeated propagation of messages around loops leads to incorrect corroboration of hypotheses; by censoring messages, this corroboration is attenuated. The communication-fusion performance tradeoff, a

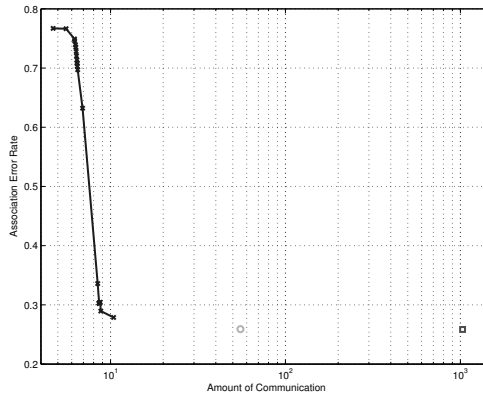


Figure 4 Performance-communication trade-off with varying thresholds for message censoring. Inference performance is evaluated by the association error rate, which is defined as the ratio of the number of measurements that are assigned to wrong targets to the total number of measurements. The amount of communication is defined as the number of messages sent by each node on average. Max-product (cyan) and TRMP (red) are plotted for comparison. (Reproduced by permission of ©2006 IEEE)

topic to which we will also return in Section 0.5, is examined more thoroughly in the next section.

0.4.2 Trading Off Accuracy for Bits in Particle-Based Messaging

The NBP algorithm for inference in graphical models involves exchanging particle-based representations of messages between nodes of the graph. When these nodes correspond to separate sensors in a network, this raises a number of questions, including (a) how does one efficiently represent and transmit these messages; and (b) how many particles are required—i.e., how can one decide what level of accuracy is required to represent the true, continuous message, and how can one send particles that provide that level of accuracy?

The heart of the first question is the following. We would like to transmit a probability distribution $q(x)$ from one sensor to another, where $q(x)$ is represented by a collection of particles $\{x_i\}$ which can be viewed as a set of i.i.d. samples from $q(x)$. Although this may seem to be a standard problem in communications and information theory, there is a key distinction—we have no interest in preserving the *order* of the samples, nor even truly in the accuracy of the transmission of those individual particles; our interest is only in obtaining an accurate reconstruction of $q(x)$ at the receiver. The fact that the set representation is order-invariant immediately suggests communications protocols which could be used to reduce the number of bits required.

For example, consider a one-dimensional distribution $q(x)$, so that the x_i are scalar values. In this case, one canonical ordering of the samples is from smallest to largest. Both sender and receiver can apply the knowledge that each subsequent particle will be larger than the previous particle to save a considerable number of bits. Indeed, in this case the optimal

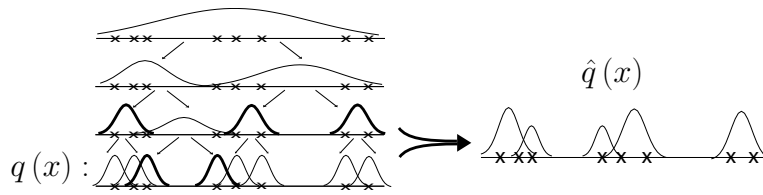


Figure 5 Approximating a message (or density estimate) $q(x)$ using a hierarchical, KD-tree structure. The same hierarchical structure can also be used to encode the approximation, providing a trade-off between bits and approximation error. (Reproduced by permission of ©2006 IEEE)

communications rate is defined not by the entropy of $q(x)$ but by the entropy of its order statistics, saving approximately a constant number of bits *per sample* (Ihler et al. 2004).

Another possible technique for encoding (either scalar or vector) particles to obtain these savings is to employ the same KD-tree structure used in NBP to provide a multi-resolution communications protocol (see Figure 5). Recall that a KD-tree provides a hierarchical clustering of the particles $\{x_i\}$ into a binary tree structure, starting from a root node (representing all the particles) and refined at each level by splitting the cluster into two subsets, corresponding to the particles to the left and right of their median value along one of the dimensions. We may also use this data structure to create a multi-resolution representation of the distribution $q(x)$ by creating simple density approximations to the clusters at each node of the KD-tree.

These representations can then be used to trade off between message accuracy and the representation cost of the message in bits. In particular, by selecting any cut through this tree, we obtain a mixture approximation $\hat{q}(x)$ to the finest-scale distribution $q(x)$. Moreover, the tree structure allows us to efficiently estimate the KL-divergence between $q(x)$ and any of these approximations. We can also use the tree structure to obtain an efficient method of transmitting the distribution at any particular resolution, by transmitting the mean and variances of its parent and then applying the knowledge (also available at the receiver due to the protocol used) that the left-child distribution will have a mean smaller than the parent mean, and the right-child will have a mean that is larger. A simple predictive encoder can be used to capture this information (for example, the left or right half of the Gaussian distribution associated with the parent node). This allows the transmitter to easily compute the cost, in bits, of sending any particular approximation.

The hierarchical structure of approximations within the KD-tree allows us to adaptively trade off quality versus communications cost. For example, a maximum allowable message distortion can be used to determine a specific cut through the tree (and corresponding message approximation), which in turn leads to a protocol for efficiently communicating that approximation. Conversely, a maximum allowable communications cost can be used to determine the most accurate approximation whose representation cost is less than the bound.

Figure 6 illustrates the performance of this method in the context of the sensor localization problem. The messages and beliefs computed during localization can be multi-modal, particularly at the initial stages of the algorithm when little information has been exchanged. However, as the iterations of sum-product proceed, more sensor locations are resolved and fewer messages consist of multi-modal distributions. As a result, an adaptive algorithm for

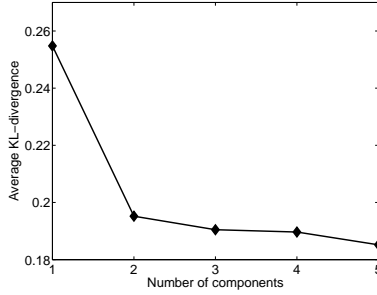


Figure 6 Reducing communications via message approximation in the sensor localization problem. Approximating each message using a single Gaussian can cause errors since multimodalities in the messages may be obscured, but using even a few components is typically sufficient to accurately represent the information.

message communication often initially sends several mixture components, but over time may require only coarse, single-mode distributions which require fewer bits to communicate. Combining this behavior with the message censoring approach described previously provides a sensor localization algorithm which is resource-aware, and trades off the cost of communications with the quality and quantity of messages sent.

0.5 The Effects of Message Approximation

The previous section described a number of ways in which limited communications resources can be conserved by allowing certain approximations or errors to occur in the sum-product algorithm, whether from censoring (using the old version of a message in place of a new one) or from explicitly approximating particle-based messages. However, the metric we care most about is not the individual message quality, but rather the accuracy of the final, fused estimates having used those approximate messages. The relationship between message approximations and the ultimate errors in the estimated beliefs is examined in detail in (Ihler et al. 2005a).

The analysis described in (Ihler et al. 2005a) considers two possible metrics for quantifying the difference between an exact and an approximate message. One of these metrics is the Kullback–Leibler divergence (as used in Section 0.4.1); the other is a measure of the “dynamic range” in this difference, equivalent to a norm on the log-messages. In particular, the dynamic range is

$$d(M_{ts}, \hat{M}_{ts}) = \sup_{x, x'} \left(\frac{M_{ts}(x) \hat{M}_{ts}(x')}{\hat{M}_{ts}(x) M_{ts}(x')} \right)^{1/2} \quad (10)$$

which can be shown to be equivalent (Ihler et al. 2005a), in the log-domain, to

$$\log d(M_{ts}, \hat{M}_{ts}) = \inf_{\alpha} \sup_x |\log \alpha + \log M_{ts}(x) - \log \hat{M}_{ts}(x)| \quad (11)$$

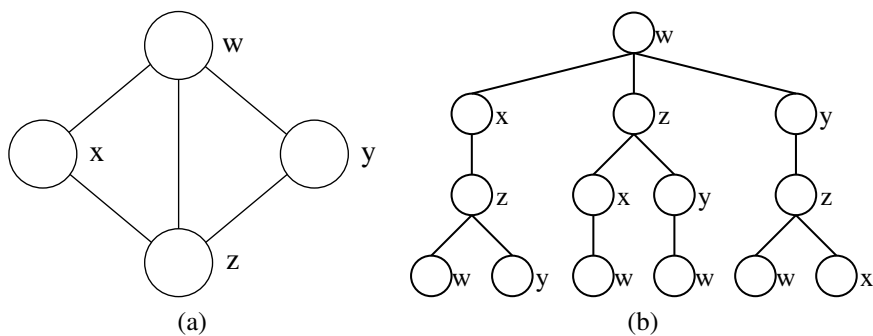


Figure 7 (a) A small graphical model and (b) the associated computation tree with root 1. The messages received at node 1 after 3 iterations of loopy sum-product in the original graph is equivalent to the messages received at the root node of the tree after 3 (or more) iterations. (Reproduced by permission of ©2006 IEEE)

The measure $d(\cdot)$ has several very important properties. First, as with KLD, it is insensitive to the (irrelevant) scaling of entire messages. Indeed, $\log d(\cdot)$ turns out to be a sup-norm on the quotient space defined by this rescaling, and can be related to the usual sup-norm between two log-messages. However, the most important property for our purposes is that the effect of errors measured by the dynamic range can be bounded through each of the two steps of the sum-product algorithm, (5). Specifically, one can show that $\log d(\cdot)$ behaves subadditively with respect to the “product” operation, so that the log of dynamic range of the product of several approximate messages is at most the sum of the logs of the dynamic ranges of the messages in the product. Furthermore, the dynamic range allows one to establish a minimum rate for the mixing produced by the “sum” operation, i.e., the convolution of the product of incoming messages with the potential function relating one node to another. This mixing behavior causes the sum operation to act as a contraction, attenuating the total error in the outgoing message. Specifically, if one measures the strength of a potential ψ_{ts} as

$$S(\psi_{ts}) = \sup_{a,b,c,d} \sup_{\psi_s, \psi_t} \frac{\psi_{ts}(a,b) \psi_t(c) \psi_s(d)}{\psi_t(a) \psi_s(b) \psi_{ts}(c,d)}$$

and denotes the product of incoming messages and local potential as M , one may show that

$$d\left(\sum \psi_{ts} M, \sum \psi_{ts} \hat{M}\right) \leq \frac{S(\psi_{ts}) d(M, \hat{M}) + 1}{S(\psi_{ts}) + d(M, \hat{M})}. \quad (12)$$

Combining (12) with the subadditivity relationship between M and its component incoming messages provides bounds on how errors propagate at each iteration of the sum-product algorithm. The behavior over multiple iterations can then be analyzed as well, and is most easily visualized via a computation tree. Figure 7 shows the computation tree for a simple graphical model with cycles; the messages that a particular node receives after several iterations involve a large number of paths, many of which may include loops. In this way, it is possible to compute a bound on the log dynamic range after any number of iterations of

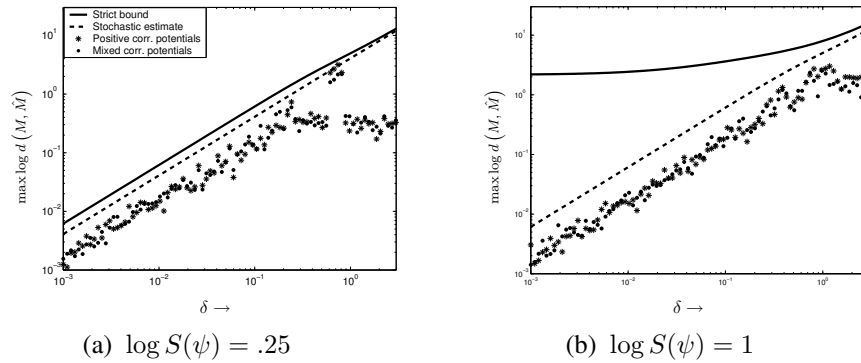


Figure 8 Maximum errors incurred as a function of the quantization error δ in a quantized implementation of sum-product. The scatterplot indicates the maximum error measured in the graph for each of 200 Monte Carlo runs, as compared to our upper bound (solid) and approximation (dashed).

(Reproduced by permission of ©2006 IEEE)

sum-product. Interestingly, this analysis not only yields bounds on behavior in the presence of message approximations, but also provides important results for sum-product in general, such as the best conditions known to date for algorithm convergence, and bounds on the distance between any two fixed points of sum-product.

Since the analysis just described yields bounds, there are cases in which the results it provides can be conservative. As an alternative, we have also developed an approximation (not a bound) built on the same principle as that used in roundoff noise analysis of digital filters, namely that at each point at which messages are approximated, the log dynamic range error can be modeled as a white noise perturbation. This assumption can then be used to provide easily computed approximations to the variance in message errors.

Figure 8 depicts the results of these analyses for two problems in which errors are introduced to sum-product via message quantization. The first problem involves relatively weak edge potentials (ones for which convergence of sum-product is guaranteed); the second involves stronger edge potentials (typical of cases in which one would expect our strict bounds to be conservative). These curves depict resulting error measures (bound, stochastic approximation, and results of simulations) as functions of the quantization error introduced at each iteration of sum-product. Note that even in the strong-potentials case the approximation generally provides an accurate (and still slightly conservative) estimate of resulting error. Most importantly, when these results are combined with methods for balancing communication costs and message approximation error, we can create a complete “audit trail” from bits used to message errors incurred and finally to resulting errors in the final estimated beliefs.

0.6 Optimizing the Use of Constrained Resources in Network Fusion

The developments in the preceding sections have provided a picture for how the message passing structure of inference algorithms on graphical models can provide the basis for effective distributed network fusion algorithms. Of course this is not the entire story: in most if not all sensor network applications, we must also deal with limitations on available *resources* (for sensing, communication, and computation). Section 0.5 provided a first look at the implications of these limitations in terms of the errors that arise from the fact that messages cannot be sent perfectly and, in some cases, may not be sent at all. While that analysis is important in understanding communication-performance tradeoffs, it does not address the problem of making optimum use of the available resources so that the network, as a whole, can achieve its overall (perhaps distributed) *objectives*.

Consideration of constrained resources highlights the fact that there are generally two sides to communication in fusion networks. The first, which is the only one present in standard message-passing algorithms, is *information push*: the transmitting node sends bits it decides are important. The second is *information pull*: the receiving node lets the transmitting node know what bits would be of most value. Furthermore, “information pull” methods, in which a processor with access to a sufficient statistic for past observations decides which resources to activate, may provide some advantage over “information push” methods, in which local sensors with access to only their own observations decide when to transmit their information.

In this section we take a look at two research directions that are aimed precisely at viewing a sensor network as a resource-constrained *team* and developing strategies for optimal coordinated use of the team’s resources. The first of the two directions focuses on distributed target tracking. As we saw in Section 0.3.2, there are systematic methods that allow us to transform data association and tracking problems into message-passing algorithms for graphical models. Here, we generalize these ideas to include two additional constraints. The first is simply that there is a power cost to sensing and computation. The second issue is that, as can be seen in Figure 3, the graphical models that arise in data association and tracking problems have some nodes that correspond to sensors but others that correspond to other “hidden” variables (regions, target states, etc.). Of course those computations need to be performed at one of the sensing nodes, and this raises the question of who takes the lead—i.e., to whom are the available measurements sent for fusion with the current estimates of target states. However, there is also a power cost in communicating which raises two of its own issues, both related to the fact that communications power required increases with distance. The first is the obvious one, namely there is a distance-related cost in transmitting raw measurements from sensor to leader. The second is more involved but at least as important. Specifically, as a target moves, there are changes in which sensors are the best to task, and when this happens, the cost of communication also requires that we consider changing the choice of leader, i.e. that we consider the problem of handoff of responsibilities for maintenance of target state estimates. Of course there is a cost for this, namely the cost of transmitting the current target information state—e.g., in particle form using the methods described in Section IV-B—from the old leader to the new. The development of a dynamic resource allocation method for balancing all of these choices and objectives—track accuracy, power conservation, choices of nodes to

task for sensing and decisions on when and to whom to hand off leader responsibilities is the subject of Section 0.6.1.

The methods of decentralized team theory provide a natural setting to investigate a push-pull notion of *self-organization* as an optimal equilibrium strategy for communications-constrained network fusion. This second line of inquiry, described in Subsection 0.6.2, takes as the starting point the fact that there are distributed objectives and communication constraints in a sensor network, and seeks to develop message-passing strategies to optimize those objectives subject to those constraints. In so doing, the network addresses and achieves *self-organization*, in which each node learns how to interpret the bits received (or perhaps *not* received) from other nodes and how to generate bits that are most informative to other nodes. Moreover, the iterative computation we describe to find such equilibria is itself a distributed message-passing algorithm to be executed “offline” (i.e., before actual measurements are processed): at each stage in the iteration, one node adjusts its local decision rule (for subsequent “online” processing) based on incoming messages from its neighbors and, in turn, sends adjusted outgoing messages to its neighbors. Some of the offline messages received by each node define, in the context of its local objectives (e.g., target detection, cost of communication), a statistical model for the information it may receive online (e.g., “what do my neighbor’s bits mean to me”), while the other offline messages define, in the context of all other nodes’ objectives, the value of information it may transmit online (e.g., “what do my bits mean to my neighbors, their neighbors, and so on”). The result of these offline iterations can be viewed as a *fusion protocol* to maintain online decision-making performance in the face of anticipated network resource constraints, emphasizing a critical point for the practitioner: if such self-organization is expected to be done in situ, as is often discussed for ad-hoc sensor networks, it must of necessity expend only a small fraction of the available power at each node.

0.6.1 Resource management for object tracking in sensor networks

Here, we consider an object tracking problem in which we seek to trade off estimation performance with energy consumed by sensing and communication. We approach the trade off between these two quantities by maximizing estimation performance subject to a constraint on energy cost, or the dual of this, i.e., minimize energy cost subject to a constraint on estimation performance. We assign to each operation (sensing, communication, etc) an energy cost, and then we seek to develop a mechanism which will allow us to choose only those actions for which the resulting estimation gain received outweighs the energy cost incurred. We assume that sensing and communication are orders of magnitude more costly than computation, and hence that the cost of computation can be neglected.

In order to operate in this regime, we must be able to calculate both the benefit of a particular action in terms of estimation performance, and the corresponding energy cost, without consuming significant communication energy. Indeed, needing to communicate in order to determine how best to use our communication resources would represent quite a conundrum. This requirement effectively dictates that we should maintain the PDF for the object under track at a single node in the network at each time step, as proposed by previous authors (e.g., (Jones et al. 2002; Liu et al. 2003)). By ensuring that all of the information which comprises this PDF is located at a single node, we can calculate future expectations without expending energy. In the context of single object tracking, we refer to this node as

the leader node. Of course, the leader node may change at each time step; this is yet another action over which we plan, taking into account the corresponding communication cost.

Entropy is a measure of uncertainty which has been applied to a wide variety of contexts including sensor management (e.g., (Chu et al. 2002; Zhao et al. 2002)). We measure the reward of obtaining a particular measurement, y_k , as the reduction in entropy that it yields in the quantity being estimated, x_k (e.g. the position and velocity of the object under track). This can be expressed equivalently as the mutual information between the observation and the quantity being estimated, conditioned on all previous measurements, $y_{0:k-1}$:

$$I(x_k; y_k | y_{0:k-1}) = H(x_k | y_{0:k-1}) - H(x_k | y_{0:k})$$

or, equivalently, the Kullback-Leibler distance between the prior and posterior distributions of the quantity being estimated, $D(p(x_k | y_{0:k}) || p(x_k | y_{0:k-1}))$.

We assume that the energy cost of each operation is known by the leader node. The specification of these abstract energy costs allows great flexibility. For example, one could choose to implement a protocol in which the tasked sensor sends the measurement, after which the leader node confirms reception. If confirmation is not received, then the tasked sensor could re-send the measurement. The communication costs could easily be incorporated into the expected cost of operating this protocol. Similarly, one could design a protocol in which the information is sent once. One could then discount the expected information reward by the probability that the measurement will never be received. One could also envision a system which adaptively selects online between these two protocols, trading off the benefits and costs of each.

When one makes resource management decisions for sensor network object tracking, one is obliged to consider not only the instantaneous benefit of decisions at the current time, but also the long-term impact of the decisions. For example, suppose that one has the choice of either immediately obtaining information about an object from a distant sensor for which communication comes at a high price, or to wait a few time steps and then obtain essentially the same information from a nearer sensor for which communication is comparatively cheap. Clearly, the knowledge of future opportunities should impact the current decisions. Furthermore, each time one makes a decision and receives the resulting measurements, one then has the opportunity to alter subsequent decisions.

The optimal solution of this category of problem is a dynamic program (e.g., (Bertsekas 2000)). Unfortunately, since our observations are noise-corrupted measurements of portions of the quantities we are estimating (e.g., position and velocity of the object being tracked), the decision state of the dynamic program will incorporate the conditional PDF of these quantities, $p(x_k | y_{0:k-1})$, a structure similar to a partially observed Markov decision process. The complexity of this state space necessitates the use of suboptimal solution methods; the optimal method requires infinite computation and storage, and direct approximations thereof have exponential complexity.

A common sub-optimal control method used for situations involving partially observed processes is Open Loop Feedback Control (OLFC). In this scheme, at each time the controller constructs a plan of which action to choose at each step in the planning horizon. In doing so, the controller does not anticipate availability of any new information while the plan is being executed. After executing one or more steps of that plan, the controller then constructs a new plan which incorporates the new information received in the interim. We utilize OLFC in

this way, at each step planning over the following N steps in time (referred to as a rolling horizon).

Greedy heuristics have shown to perform well in many object tracking applications. These methods select actions which maximize the instantaneous reward (e.g. the largest reduction in entropy), paying no regard to future opportunities. Indeed, if estimation performance as measured by mutual information is the only criterion, then one can prove that, in open loop, greedy methods produce a result with reward no less than one half that of the optimal solution (see Krause and Guestrin (2005); Williams et al. (2006a)). In the context of sensor networks this situation changes. If we operate the same greedy algorithm paying attention only to short-term information gain, the energy consumption may be arbitrarily high. Conversely, if we operate the a greedy algorithm paying attention only to the immediate communication cost, the estimation performance may be arbitrarily poor. How one should modify this heuristic in order to *optimally* trade off these competing objectives is an open question.

We formulate the trade off between estimation performance and energy consumption through a constrained dynamic program, similarly to (Castañon 1997), where we either maximize estimation performance subject to a constraint on energy usage, or minimize energy usage subject to a constraint on estimation performance. Denoting the control policy for the next N steps as $\pi = \{\mu_k, \dots, \mu_{k+N-1}\}$, and the decision state (i.e., the combination of conditional PDF of object state and the previous leader node) as \mathbb{X}_k the underlying dynamic program becomes:

$$\begin{aligned} \min_{\pi} \mathbb{E} \left[\sum_{i=k}^{k+N-1} g(\mathbb{X}_i, \mu_i(\mathbb{X}_i)) \right] \\ \text{s.t. } \mathbb{E} \left[\sum_{i=k}^{k+N-1} G(\mathbb{X}_i, \mu_i(\mathbb{X}_i)) \right] \leq M \end{aligned} \quad (13)$$

In the first formulation, which optimizes estimation performance subject to a constraint on energy cost, we set the per-stage cost to be

$$g(\mathbb{X}_k, u_k) = -I(x_k; y_k^{u_k} | \mathbb{X}_k) \quad (14)$$

so that the sum of the rewards is the total reduction in entropy over the planning horizon. We set the per-stage constraint contribution $G(\mathbb{X}_k, u_k)$ to be the energy cost of implementing the decisions u_k . In the dual formulation, which minimizes energy cost subject to a constraint on estimation performance, these two quantities are reversed.

By approaching the constrained optimization using a Lagrangian relaxation (a common approximation for integer programming problems), we find a principled method for trading off competing objectives using algorithms based on an extension of the basic greedy heuristic. Our approach exploits the additive structure of both information rewards (e.g., due to the additive decomposition of mutual information) and energy costs in order to find a per-stage cost of the form:

$$\bar{g}(\mathbb{X}_k, u_k, \lambda) = g(\mathbb{X}_k, u_k) + \lambda G(\mathbb{X}_k, u_k) \quad (15)$$

This can be motivated by applying a Lagrangian relaxation, defining the dual function:

$$J_k^D(\mathbb{X}_k, \lambda) = \min_{\pi} \mathbb{E} \left[\sum_{i=k}^{k+N-1} g(\mathbb{X}_i, \mu_i(\mathbb{X}_i)) + \lambda \left(\sum_{i=k}^{k+N-1} G(\mathbb{X}_i, \mu_i(\mathbb{X}_i)) - M \right) \right] \quad (16)$$

and solving the dual optimization problem involving this function:

$$J_k^L(\mathbb{X}_k) = \max_{\lambda \geq 0} J_k^D(\mathbb{X}_k \lambda) \quad (17)$$

We note that the dual function $J_k^D(\mathbb{X}_k, \lambda)$ takes the form of an unconstrained dynamic program with a modified per-stage cost, defined in (15). This motivates the use of an algorithm that is greedy with respect to this augmented cost, in which we choose to utilize a sensor only if the expected information of the sensor measurement outweighs the cost of obtaining the measurement. For example, suppose we want to select a subset of sensors to activate at a particular time step during operation. We proceed with greedy selection using this augmented cost, activating the sensor with the smallest augmented cost. When either all sensors are active, or all augmented costs are positive (indicating that, conditioned on the observations already chosen, the additional benefit of any remaining observation does not outweigh the cost of obtaining it), we terminate with the current subset.

In the context of sensor networks, in which communication is a primary source of energy expenditure, the cost of obtaining an observation is dependent upon the current choice of leader node, to which the sensor taking the measurement must transmit the observation. Conditioned on a particular choice of leader node trajectory (i.e., a choice of which node to select as leader at each time step in the planning horizon), the subset selection method above provides a means of selecting which sensors to activate at each time. However, in online operation, we simultaneously choose which sensors to activate, *and* which node to activate as leader. Our approach to this problem is a hierarchical decomposition, in which we consider different choices of leader node at each time, and then conditioned on these choices, we use the greedy subset selection. Obviously we cannot plan over all possible trajectories of leader node, so we utilize a scheme which adaptively prunes the search tree, at each level comparing the sequences ending with a particular node as leader and keeping the sequence with highest reward. A detailed description of our scheme can be found in Williams et al. (2005, 2006b).

The final question which must be addressed is how to choose the value of the Lagrange multiplier. This could be performed using a number of methods such as a line search or a subgradient search. However, in practice, we simply update the value once at each iteration of the algorithm depending on whether the constraint had slack or was exceeded. If the constraint was exceeded, then the Lagrange multiplier is increased (either additively or multiplicatively by a constant amount); if the constraint had slack then the value is decreased. This provides an online approximation of a subgradient search.

In essence, the algorithm described performs long-term planning over trajectories of leader node and Lagrange multiplier value, and short-term planning over which subset of sensors to activate conditioned on the long-term choice of leader node and Lagrange multiplier. In this way, we gain the efficiency of the greedy heuristic while still capturing the trade off between the competing quantities of estimation performance and energy cost.

As an example, Figure 9 illustrates the performance of our method in which we track an object moving through a network of 20 randomly-placed acoustic sensors. The diagram demonstrates that the communication-constrained formulation provides a way of controlling sensor selection and leader node which reduces the communication cost and improves estimation performance substantially over the myopic single-sensor methods, which at each time activate and select as leader node the sensor with the measurement producing the largest expected reduction in entropy. The information-constrained formulation allows for an additional saving in communication cost while meeting an estimation criterion wherever possible.

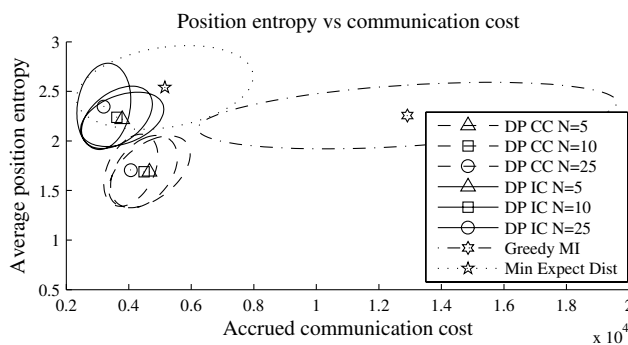


Figure 9 Position entropy versus communication cost for a source tracking problem using a network of 20 acoustic sensors, using several different tracking algorithms. Each ellipse shows the mean (center) and covariance of the communication cost and position entropy, obtained by averaging over 100 Monte-Carlo runs. The algorithms considered are: dynamic programming method with communication constraint (DP CC) and information constraint (DP IC) with different planning horizon lengths (N); and two greedy methods in which the leader node at each iteration is selected as the node with either the largest mutual information (greedy MI) or the smallest expected square distance to the object (min expect dist).

The previous discussion has been in the context of tracking a single object. If multiple objects are moving independently and are never observed by the same sensor, then the algorithm can be executed in parallel for each object. If objects become close together and are observed by the same sensor, then the joint observation process induces conditional dependency between the objects. In this situation, one could choose to store the joint pdf of the two objects at a single node, utilizing a single instance of the algorithm previously described for joint estimation and planning for the two objects. Thereafter, at each step one could consider the control option to discard the joint representation in favor of two marginal pdfs, evaluating the information loss which would result against the communication saving it would yield. Alternatively, one could utilize a distributed representation such as that described in Section 0.3 with some prediction of the communication costs that will be necessary.

0.6.2 Distributed Inference with Severe Communication Constraints

The previous analyses demonstrate fundamental tradeoffs between achievable fusion performance and available communication resources in distributed sensor networks. The results in Section 0.4, in particular, establish that distributed message-passing algorithms for graphical models are robust to substantial message errors, but also suggest performance may be unsatisfactory as communication resources become severely constrained. For example, Figure 4 illustrates a catastrophic failure in data association performance when the censoring thresholds result in a very low amount of communication. Also observe that the upper bound and approximation curves in Figure 8 tend to a common positive slope as per-link quantization error becomes arbitrarily large, implying very low link capacities (i.e., reliable communication of only a few bits per desired estimate) which impact performance. That there are limits

to the demonstrated robustness of conventional message-passing algorithms is not surprising, considering they are originally derived assuming ideal communications. In this subsection, we explicitly model the presence of low-rate, unreliable communication links and examine the extent to which message-passing algorithms other than those discussed in Section 0.2 can mitigate the potential loss in fusion performance.

Problem Formulation

We begin with a graphical model (as described in Section 0.2) and focus on the fusion objective of estimating a discrete state process $X = \{X_s, s \in V\}$, based on a noisy measurement process $Y = \{Y_s, s \in V\}$. Our goal is to design a distributed fusion approach which minimizes the expected number of errors across all inference nodes, X (i.e., bit-error-rate in the case of binary state variables). Specifically, we denote by Γ the set of all functions γ that map the support of Y into the support of X , we seek the estimator $\hat{X} = \bar{\gamma}^*(Y)$ such that

$$J(\bar{\gamma}^*) = \min_{\gamma \in \Gamma} \underbrace{E[c(\gamma(Y), X)]}_{\equiv J(\gamma)}. \quad (18)$$

Here, so that J indeed measures the expected number of nodes in error, the numeric ‘‘cost’’ $c(\hat{x}, x)$ associated to each possible realization of the joint process (\hat{X}, X) is taken to be

$$c(\hat{x}, x) = \sum_{s \in V} c(\hat{x}_s, x_s) \quad \text{where, for each node } s, \quad c(\hat{x}_s, x_s) = \begin{cases} 0 & , \hat{x}_s = x_s \\ 1 & , \hat{x}_s \neq x_s \end{cases}. \quad (19)$$

Classical decision theory (Van Trees 1968) states that the estimator satisfying (18) for the costs in (19) is equivalent to finding the mode of every state variable’s marginal distribution conditioned on *all* measurements i.e., per realization $Y = y$, compute $\hat{x} = \{\hat{x}_s, s \in V\}$ via

$$\hat{x}_s = \arg \max_{x_s} p(x_s | y), \quad \forall s \in V. \quad (20)$$

Indeed, as discussed in previous sections, the goal of sum–product algorithms is to obtain exactly these marginal distributions $p(x_s | y)$ at every node $s \in V$. It follows that the realization of each optimal estimate $\hat{x} = \bar{\gamma}^*(y)$ requires total communication overhead of *at least* $2|E|$ real-valued messages. In contrast, we embark upon realizing each estimate $\hat{x} = \gamma(y)$ under the restriction that total communication overhead is *at most* $|E|$ finite-alphabet messages, or *symbols* (e.g., just one bit per edge). Such severe constraints clearly render the optimal estimator in (20) infeasible, even for tree-structured graphical models, shifting the purpose of our problem formulation to finding a feasible (yet effective) estimator.

We view restrictions on communication as placing explicit constraints on the function space Γ in (18). Specifically, let the *network topology* be defined by a directed acyclic graph $N = (V, D)$. Assume each edge $(t, s) \in D$ represents a point-to-point communication link from node t to node s with known finite capacity (in bits per estimate), denoting by

$$\pi(s) = \{t \in V, (t, s) \in D\} \quad \text{and} \quad \chi(s) = \{t \in V, (s, t) \in D\}$$

the *parents* and *children*, respectively, of each node s . Sources of unreliable communication (e.g., multicast interference, dropped packets) at node s are modeled by a discrete-memoryless channel $p(z_s | x_s, m_{\pi(s) \rightarrow s})$: here, process Z_s denotes the information received

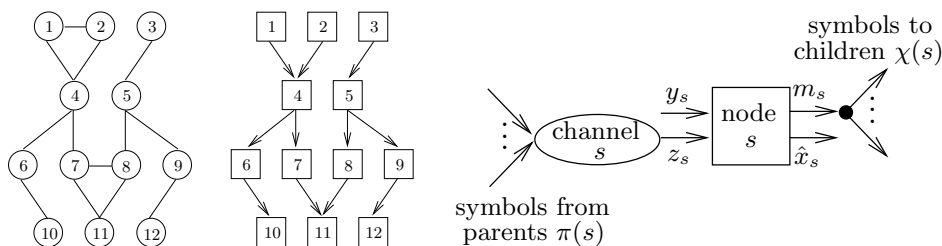


Figure 10 The two types of graphs in our formulation: (a) an undirected graph G underlying a probabilistic graphical model, and (b) a directed acyclic graph N , defining the parents $\pi(s)$ and children $\chi(s)$ of each node s in a network topology, along with (c) the implied fusion rule at each node, invoked in succession from parent-less nodes to child-less nodes in N .

by node s given the symbols $m_{\pi(s) \rightarrow s} = \{m_{t \rightarrow s}, t \in \pi(s)\}$ were transmitted to node s (see Figure 10). Moreover, fusion objectives with costly or selective (i.e., censored) communication can be captured by augmenting the cost function $c(\hat{x}, x)$ to also depend on all transmitted symbols $m = \{m_{s \rightarrow t}, (s, t) \in D\}$. For example, suppose the alphabet of each symbol $m_{s \rightarrow t}$ includes a “no-send” option and a numeric cost $c(m_s)$ equals the number of such symbols $m_s = \{m_{s \rightarrow t}, t \in \chi(s)\}$ selected by node s that do *not* exercise this option: then, choosing

$$c(m, \hat{x}, x) = \sum_{s \in V} c(\hat{x}_s, x_s) + \lambda c(m_s), \quad \text{for a specified constant } \lambda \geq 0, \quad (21)$$

results in J measuring a (λ -weighted) sum of the *node-error-rate* and *link-use-rate*.

Altogether, denoting by Γ_s the set of functions γ_s at node s by which any particular input (y_s, z_s) maps to an output (m_s, \hat{x}_s) , we seek the collection $\gamma^* = \{\gamma_s^*, s \in V\}$ such that

$$J(\gamma^*) = \min_{\gamma \in \Gamma} J(\gamma) \quad \text{subject to } \gamma \in \Gamma(N) = \Gamma_1 \times \cdots \times \Gamma_{|V|}. \quad (22)$$

By definition, any collection of local decision processes $(M_s, \hat{X}_s) = \gamma_s(Y_s, Z_s)$ satisfying (22) will be both feasible and achieve the minimum loss in fusion performance. This formulation consolidates a number of problem variants studied in decentralized detection (Tsitsiklis 1993; Varshney 1997; Viswanathan and Varshney 1997), including the allowance of an arbitrary network topology (Pete et al. 1996) and selective or unreliable communication (Chen et al. 2004; Papastavrou and Athans 1986; Pothiwala 1989; Rago et al. 1996).

Summary of Algorithmic Solutions

An important implication of our problem formulation is the distinction between the sensor network’s *online* algorithm, or the implementation of any feasible fusion strategy $\gamma \in \Gamma(N)$, and *offline* algorithm, or the solution to the constrained optimization problem in (22). In particular, when online estimation is severely resource-constrained, the offline optimization by which performance loss can be mitigated introduces an additional tax on network resources. Given we anticipate intermittent reorganization by the network to stay connected (due to e.g., node dropouts, link failures), we must also anticipate the incentive for intermittent re-optimization. Hence, unless the offline algorithm itself admits an efficient distributed

implementation, little hope exists for preserving satisfactory fusion performance without also rapidly diminishing the resources available for actual online measurement processing.

As highlighted earlier, the constrained optimization problem in (22) falls within the class of (discrete) decentralized decision problems, for which optimal solution is known to be NP-hard (Tsitsiklis and Athans 1985). Also known is an approximation to the problem, called person-by-person optimality in team theory (Marschak and Radner 1972), for which analytical progress can be made. It applies to our formulation provided the channel noise at every node is independent of the channel noise at all other nodes as well as the observation noise at all nodes (Kreidl and Willisky 2006a). The resulting online strategy is essentially equivalent to a collection of elementary Bayesian detectors, taking the form

$$\gamma_s^*(Y_s, Z_s) = \arg \min_{(m_s, \hat{x}_s)} \sum_{x_s} \theta_s^*(m_s, \hat{x}_s, x_s; Z_s) p(Y_s | x_s) \quad \text{with probability one,} \quad (23)$$

where parameters $\theta^* = \{\theta_s^*, s \in V\}$ (reducing to a set of likelihood-ratio thresholds in the case of binary state variables) are globally coupled through a nonlinear fixed-point equation,

$$\theta_s = f_s(\theta_{V \setminus s}) \quad \text{for } s \in V, \quad (24)$$

to be solved offline (Kreidl and Willisky 2006a,b; Tang et al. 1991; Tsitsiklis 1993). Moreover, given any initial parameters θ^0 , iterating (24) in a node-by-node fashion (i.e., Gauss-Seidel iterations) guarantees a parameter sequence $\{\theta^k\}$ for which the associated cost sequence $\{J(\gamma^k)\}$ is convergent. In general, however, a distributed implementation of this offline algorithm (i) assumes all nodes are initialized with common global knowledge i.e., probabilities $p(x)$, costs $c(m, \hat{x}, x)$ and channels $\{p(z_s | x_s, m_{\pi(s) \rightarrow s}), s \in V\}$; and (ii) requires computation/communication overhead that scales exponentially with the number of nodes.

Provided the directed graph N is tree-structured (as in Figure 10(b)), it can be shown (Kreidl and Willisky 2006a,b; Tang et al. 1993) that (24) specializes to the form

$$\begin{aligned} P_{s \rightarrow \chi(s)} &= f_s^1(\theta_s, P_{\pi(s) \rightarrow s}) \\ \theta_s &= f_s^2(P_{\pi(s) \rightarrow s}, C_{\chi(s) \rightarrow s}) \quad \text{for } s \in V. \\ C_{s \rightarrow \pi(s)} &= f_s^3(\theta_s, P_{\pi(s) \rightarrow s}, C_{\chi(s) \rightarrow s}) \end{aligned} \quad (25)$$

Here, for each link (s, t) in N , the *forward messages* $P_{s \rightarrow t}(m_{s \rightarrow t} | x_s)$ represent a likelihood function, giving global context to the symbol $m_{s \rightarrow t}$ from the receiving node's perspective, while the *backward messages* $C_{t \rightarrow s}(m_{s \rightarrow t}, x_s)$ represent an expected cost-to-go function, giving global context to the symbol $m_{s \rightarrow t}$ from the transmitting node's perspective. Observe that parameters θ are still globally coupled, but this coupling is now expressed only implicitly through the forward and backward message recursions. Indeed, convergence of the sequence $\{J(\gamma^k)\}$ is guaranteed by iterating the equations of (25) in any order corresponding to repeated forward-backward sweeps through the network (Kreidl and Willisky 2006a,b). Moreover, a distributed implementation of this offline message-passing algorithm (i) assumes each node s is initialized with local knowledge i.e., probabilities $p(x_{\pi(s)}, x_s)$, costs $c(m_s, \hat{x}_s, x_s)$ and channel $p(z_s | x_s, m_{\pi(s) \rightarrow s})$; and (ii) requires computation/communication overhead that scales linearly with the number of nodes. Note that this local message-passing algorithm can always be applied to arbitrary directed acyclic networks, but convergence is then in question.

An Illustrative Example

Consider the following instance of the problem described above, involving only three main parameters for ease of illustration. The joint distribution $p(x, y)$ is defined on the graph G shown in Figure 10(a): each node's state variable x_s and measurement variable y_s is binary-valued and (scalar) real-valued, respectively, where potential functions are taken to be

$$\psi_{st}(x_s, x_t) = \begin{cases} \eta & , \quad x_s = x_t \\ 1 - \eta & , \quad x_s \neq x_t \end{cases} \quad \text{and} \quad \psi_s(x_s, y_s) = \exp\left(-\frac{(y_s - (x_s - 0.5))^2}{2\sigma^2}\right).$$

The parameter $\eta \in (0, 1)$ captures the correlation between neighboring states (i.e., negative or positive for $\eta < 0.5$ or $\eta > 0.5$, respectively) while the parameter $\sigma \in (0, \infty)$ captures the measurement accuracy at every node. The network topology is the graph N shown in Figure 10(b) with each link taken to be an independent and identically distributed binary erasure channel together with a (reliable) "no-send" option: each channel model is $p(z_s | x_s, m_{\pi(s) \rightarrow s}) = \prod_{t \in \pi(s)} p(z_{t \rightarrow s} | m_{t \rightarrow s})$ where

$$p(z_{t \rightarrow s} | m_{t \rightarrow s}) = \begin{cases} 1 - \epsilon & , \quad z_{t \rightarrow s} = m_{t \rightarrow s} \text{ when } m_{t \rightarrow s} \in \{0, 1\} \\ \epsilon & , \quad z_{t \rightarrow s} = \emptyset \text{ when } m_{t \rightarrow s} \in \{0, 1\} \\ 1 & , \quad z_{t \rightarrow s} = \emptyset \text{ when } m_{t \rightarrow s} = \emptyset \\ 0 & , \quad \text{otherwise} \end{cases}.$$

The parameter $\epsilon \in [0, 1]$ captures the reliability of the unit-rate communication network (i.e., perfect for $\epsilon = 0$), and the correct symbol is always received in the absence of an erasure.

We consider the cost function $c(m, \hat{x}, x)$ defined in (21), varying parameter $\lambda \geq 0$ in small increments (of size 3×10^{-4}) and repeatedly applying the offline message-passing algorithm until the sequence $\{J(\gamma^k)\}$ converges (using a tolerance of 10^{-3}). The initial parameters θ_s^0 at each node s are chosen to partition the local measurement space into $6|\chi(s)|$ intervals, assigning

- (i) decision $\hat{x}_s = m_{s \rightarrow t} = 0$ to the interval $(-\infty, (1 - 3|\chi(s)|)\sigma^{-1})$
- (ii) decision $\hat{x}_s = m_{s \rightarrow t} = 1$ to the interval $(\sigma^{-1}(3|\chi(s)| - 1), \infty)$, and
- (iii) all remaining decisions to equally-spaced sub-intervals of $[\sigma^{-1}(1 - 3|\chi(s)|), (3|\chi(s)| - 1)\sigma^{-1}]$.

Results are shown in Figure 11 and indicate that the optimized team strategy consistently hedges against all sources of uncertainty as we weigh between fusion inaccuracy (i.e., node-error-rate) and online communication overhead (i.e., link-use-rate). Table 1 indicates that offline convergence occurs in roughly five iterations, each such iteration assuming the reliable transmission of $12|D| = 132$ real numbers. Thus, while satisfactory fusion performance is achievable given online communication can be at most $|D| = 11$ bits (per estimate), it depends upon offline communication on the order of 660 real-valued messages (per reorganization).

0.7 Discussion

In this chapter we have described a collection of research results and directions that build on the evocative connection between distributed fusion in sensor networks and message-passing algorithms in graphical models. Through two example applications (self-localization

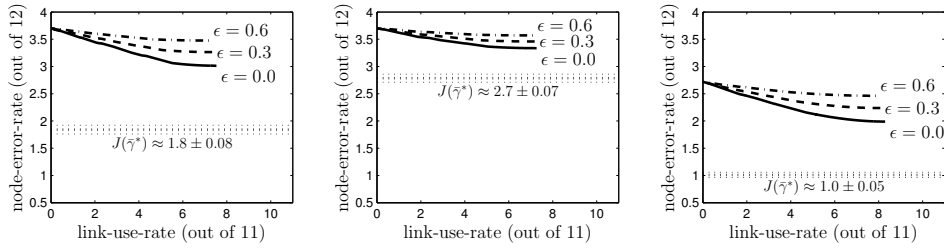


Figure 11 Optimized tradeoff curves, assuming the graphs shown in Figure 10, as a function of link unreliability (i.e., parameter ϵ) for (a) nominal state correlation and measurement noise, (b) low state correlation but nominal measurement noise, and (c) nominal state correlation but low measurement noise. Each curve is generated by incrementing parameter λ in (21) from zero, meaning online communication is cost-free, to a value in which the offline message-passing algorithm converges to the *myopic* strategy (i.e., each node acting as if in isolation), meaning any online communication is too costly relative to the potential improvement in node-error-rate performance. The link-use-rate is never at its maximum and the node-error-rate never exceeds that of the myopic strategy: that is, the optimized team strategy consistently regulates its exploitation of the “no-send” option (i.e., “no news is news,” even with an imperfect communication medium) and gracefully avoids catastrophic performance failure. Also shown in each plot (by the horizontal dotted lines) is a Monte-Carlo estimate (plus/minus one standard deviation) of the *unconstrained* optimal node-error-rate: observe that the offline message-passing algorithm can recover up to (a) 37%, (b) 40% and (c) 43% of the fusion performance lost by the myopic strategy.

Table 1 Overhead of message-passing algorithm to generate the curves of Figure 11.

Link Unreliability ϵ	Lowest Myopic Weight			Offline Iteration Count		
	(a)	(b)	(c)	(a)	(b)	(c)
0.0	0.1368	0.0774	0.1056	4.5 ± 0.74	4.4 ± 0.71	4.7 ± 0.65
0.3	0.0957	0.0543	0.0762	4.3 ± 0.74	4.2 ± 0.71	4.1 ± 0.29
0.6	0.0549	0.0312	0.0462	3.9 ± 0.89	3.9 ± 0.78	4.0 ± 0.08

“Lowest Myopic Weight” refers to the value of parameter λ at which convergence to the myopic strategy (always in exactly two iterations) first occurs; and “Offline Iteration Count” refers to the average (plus/minus one standard deviation) number of iterations to convergence, taken over only the samples of λ below the lowest myopic weight. Note that, all other things being equal, the Lowest Myopic Weight is inversely related to Link Unreliability: with respect to achievable fusion accuracy, the offline algorithm captures the diminishing marginal value of online communication as link reliability degrades. Two other trends are also worth noting, but whether they apply beyond these particular examples are open questions: (i) relative to the nominal conditions (i.e., column (a)), lower state correlation or lower measurement noise (i.e., column (b) or column (c), respectively) also diminish the value of online communication; and (ii) the on-average rate of offline convergence increases as (online) link reliability degrades.

and data association/target tracking) we have shown how network fusion problems can be mapped to graphical models, allowing one, in principle to apply graphical model inference methods to sensor network fusion problems. Not unexpectedly, of course, there are issues of critical importance in sensor networks-in particular related to constraints on available power (for communication and/or sensing)-that require analysis beyond that found in the graphical model literature, and we have presented two lines of inquiry aimed at addressing these issues.

The first involved the idea of simply approximating or “censoring” (i.e., not sending) messages in graphical model inference procedures. In the context of data association we described how censoring can lead to desirable adaptivity in network messaging (“only say something when you have something new to say”), and experiments show a sharp threshold in performance; i.e., there appear to exist critical communication rates in data association. Above this rate, performance varies only gradually with reduction in communication, but at this critical value, performance degrades precipitously. In the context of self-localization, in which messages represent “particles” we described a framework for trading off bits in communicating these particle-based messages against overall fusion performance. Interestingly, our methodology for mapping message approximations to fusion performance has independent value in the context of graphical models, as it provides error bounds and convergence results for belief propagation algorithms.

The second line of inquiry involved the development of methods for network fusion that optimize overall fusion performance taking into account, from the start, that there are costs or constraints on power consumption or communication. In the context of target tracking we described the fact that communication costs are incurred for two distinct reasons: Communicating sensed measurements to a leader or fusion node and the communication required to hand off leader responsibility as an object moves through a sensor field. Combining these with the power cost of sensing, we have described a framework for dynamically optimizing the tradeoff between power consumption and tracking accuracy. The second part of our work on optimizing resource utilization formulated the problem of designing message-passing strategies to achieve overall network fusion objectives subject to constraints on node-to-node communication. As we discussed, the resulting message-passing strategy for each not only takes into account power concerns but also the objectives of those receiving the messages that are transmitted, capturing not only information-push from transmitter to receiver but also information-pull from receiver to transmitter. Importantly, the algorithms to achieve this coordinated team behavior have message-passing structure themselves and point to the fact that the process of network organization itself consumes some of the resources that the resulting strategies aim to conserve.

While the results in this chapter provide important components for a methodology for distributed fusion in sensor networks, they are far closer to the start than the end of the story, and there are many important lines of research that remain to be considered, some of which are being pursued by our team. A first of these involves the idea of developing different ways of organizing message passing that may be more appropriate for sensor networks. For example, one can imagine operational structures in which “seed” nodes initiate messaging, propagating information radially outward, fusing information around these radially expanding regions as they meet, and then propagating information back inward toward the seed nodes. Such an algorithmic structure allows great flexibility (e.g., one can imagine allowing any sensor to act as a seed if it measures something of interest) and also leads to new algorithms with great promise for inference on graphical models more generally. We refer the reader to (Johnson

and Willsky 2006, in review) for a first treatment of this approach. Also, the computation tree interpretation of the sum-product algorithm allows one to clearly see the computations that sum-product fails to make that a truly optimal algorithm would—computations that in essence take into account the dependencies between messages that sum-product neglects (Johnson et al. 2005). This suggests another line of research that focuses on one of the significant differences between standard graphical model inference problems and sensor networks. In particular, when viewed as a sensor network fusion algorithm sum-product has the property that it makes very little use of local node memory and computational power (all that is remembered from step to step are the most recent messages, and all that are computed are essentially the sum-product computations). Can we develop algorithms that use more memory and perform more local computation and that as a result reduce the number of messages that need to be sent? Several ideas along these lines are currently under investigation. Also, a standard feature in wireless networks is the inclusion of header bits that provide information on the path a message has taken from initiator to receiver. Can we take advantage of such header bits to capture dependencies between messages so that they can then be used to fuse messages in a manner that is not as naïve as assuming conditional independence? Of course using such header bits for what we term informational pedigree means that there are fewer bits available for the actual message, so that the message quantization error will be larger. How does the error incurred by such an increased quantization error compare to the additional fusion accuracy provided by providing these pedigree bits? Current research building on what we have presented here, is addressing this question.

Numerous other directions for further work also suggest themselves, such as allowing nodes to actively *request* information from other nodes. Indeed if this is part of the self-organizing protocol of the sensor network, then each node not only will be able to extract information from such a request but also from the *absence* of such a request (i.e., “no news is news”). Also, while some of the work in Section 0.6 can accommodate imperfect communication channels, there is a real need to expand the development to allow for more serious disruptions-e.g., the failure of a node, which requires that message-passing strategies be robust or adaptable to such failures. This touches on an even richer class of problems that arise when we allow the possibility that the structure of the network; i.e., the messaging graph-can be different from the statistical graph underlying the inference problem being solved by the network. As questions such as these make clear, there is much more to be done in an area that cuts across the areas of signal processing, estimation, communication, computation, and optimization in new and fascinating ways.

Bibliography

- Bertsekas DP 2000 *Dynamic Programming and Optimal Control* second edn. Athena Scientific, Belmont, MA.
- Brémaud P 1991 *Markov chains, Gibbs fields, Monte Carlo simulation, and queues*. Springer.
- Castañón DA 1997 Approximate dynamic programming for sensor management *Proc 36th Conference on Decision and Control*, pp. 1202–1207. IEEE.
- Chen B, Jiang R, Kasetkasem T and Varshney PK 2004 Channel aware decision fusion in wireless sensor networks. *IEEE Trans. Signal Processing* **52**(12), 3454–3458.
- Chen L, Çetin M and Willsky AS 2005a Distributed data association for multi-target tracking in sensor networks *Information Fusion 2005*, Philadelphia, PA.
- Chen L, Wainwright M, Cetin M and Willsky A 2005b Data association based on optimization in graphical models with applications to sensor networks. *Mathematical and Computer Modeling (special issue on optimization and control for military applications)*. to appear.
- Chu M, Haussecker H and Zhao F 2002 Scalable information-driven sensor querying and routing for ad hoc heterogeneous sensor networks. *International Journal of High Performance Computing Applications* **16**(3), 293–313.
- Cowell RG, Dawid AP, Lauritzen SL and Spiegelhalter DJ 1999 *Probabilistic Networks and Expert Systems* Statistics for Engineering and Information Science. Springer-Verlag.
- Feldman J, Wainwright MJ and Karger DR 2005 Using linear programming to decode binary linear codes. *IEEE Transactions on Information Theory* **51**, 954–972.
- Forney, Jr. GD 1973 The Viterbi algorithm. *Proc. IEEE* **61**, 268–277.
- Freeman WT and Weiss Y 2001 On the optimality of solutions of the max-product belief propagation algorithm in arbitrary graphs. *IEEE Trans. Info. Theory* **47**, 736–744.
- Heskes T, Albers K and Kappen B 2003 Approximate inference and constrained optimization *Uncertainty in Artificial Intelligence*, vol. 13, pp. 313–320.
- Ihler A, Fisher III J and Willsky A 2005a Loopy belief propagation: Convergence and effects of message errors. *Journal of Machine Learning Research* **6**, 905–936.
- Ihler A, Fisher III J, Moses R and Willsky A 2005b Nonparametric belief propagation for self-localization of sensor networks. *IEEE J. on Select Areas in Communication* **23**(4), 809–819.
- Ihler AT 2005 *Inference in Sensor Networks: Graphical Models and Particle Methods* PhD thesis Massachusetts Institute of Technology.
- Ihler AT, Fisher III JW and Willsky AS 2004 Communication-constrained inference. Technical Report 2601, MIT, Laboratory for Information and Decision Systems.
- Ihler AT, Sudderth EB, Freeman WT and Willsky AS 2003 Efficient multiscale sampling from products of Gaussian mixtures *Neural Info. Proc. Systems 17*.
- Johnson J and Willsky A 2006, in review A recursive model-reduction method for approximate inference in Gaussian Markov random fields. *IEEE Trans. Image Processing*.
- Johnson J, Malioutov D and Willsky A 2005 Walk-sum interpretation and analysis of Gaussian belief propagation *Adv. Neural Inf. Proc. Systems*.

- Jones M, Mehrotra S and Park JH 2002 Tasking distributed sensor networks. *International Journal of High Performance Computing Applications* **16**(3), 243–257.
- Jordan M 1998 *Learning in Graphical Models*. MIT Press.
- Jordan M 2004 Graphical models. *Statistical Science* **19**(1), 140–155.
- Jordan M in preparation *An Introduction to Probabilistic Graphical Models*.
- Kolmogorov V and Wainwright MJ 2005 On optimality properties of tree-reweighted message-passing. *Uncertainty in Artificial Intelligence*.
- Krause A and Guestrin C 2005 Near-optimal nonmyopic value of information in graphical models *UAI 2005*.
- Kreidl OP and Willsky AS 2006a Efficient message-passing algorithms for optimizing decentralized detection networks *Proc. of the 45th IEEE Conf. on Decision and Control*. In review.
- Kreidl OP and Willsky AS 2006b Inference with minimal communication: a decision-theoretic variational approach *Adv. Neural Inf. Proc. Systems* vol. 19 MIT Press Cambridge, MA.
- Kschischang F 2003 Codes defined on graphs. *IEEE Signal Processing Magazine* **41**, 118–125.
- Kschischang F, Frey B and Loeliger HA 2001 Factor graphs and the sum-product algorithm. *IEEE Trans. Info. Theory* **47**(2), 498–519.
- Kurien T 1990 Issues in the design of practical multitarget tracking algorithms In *Multitarget-Multisensor Tracking: Advanced Applications* (ed. Bar-Shalom Y) vol. 1 Artech House Norwood, MA pp. 43–83.
- Lauritzen SL 1996 *Graphical Models*. Oxford University Press, Oxford.
- Liu J, Reich J and Zhao F 2003 Collaborative in-network processing for target tracking. *EURASIP Journal on Applied Signal Processing* (4), 378–391.
- Loeliger HA 2004 An introduction to factor graphs. *IEEE Signal Processing Magazine* **21**, 28–41.
- Marschak J and Radner R 1972 *The Economic Theory of Teams*. Yale University Press, New Haven, CT.
- Mooij JM and Kappen HJ 2005a On the properties of the Bethe approximation and loopy belief propagation on binary networks. *Journal of Statistical Mechanics: Theory and Experiment* **P11012**, 407–432.
- Mooij JM and Kappen HJ 2005b Sufficient conditions for convergence of loopy belief propagation. Technical Report arxiv.cs.IT:0504030, University of Nijmegen. Submitted to IEEE Trans. Info. Theory.
- Moses R, Krishnamurthy D and Patterson R 2003 Self-localization for wireless networks. *Eurasip Journal on Applied Signal Processing* pp. 348–358.
- Papastavrou JD and Athans M 1986 A distributed hypothesis-testing team decision problem with communications cost *Proc. of the 25th IEEE Conf. on Decision and Control*, pp. 219–225.
- Patwari N, Hero III AO, Perkins M, N. S. Correal N and O’Dea RJ 2003 Relative location estimation in wireless sensor networks. *IEEE Trans. Signal Processing* **51**(8), 2137–2148.
- Pearl J 1988 *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufman, San Mateo.
- Pete A, Pattipati KR and Kleinman DL 1996 Optimization of decision networks in structured task environments. *IEEE Trans. Systems, Man and Cybernetics* **26**(6), 739–748.
- Poithiawala J 1989 *Analysis of a two-sensor tandem distributed detection network* Master’s thesis MIT Department of Electrical Engineering and Computer Science Cambridge, MA.
- Rago C, Willett P and Bar-Shalom Y 1996 Censoring sensors: A low-communication-rate scheme for distributed detection. *IEEE Trans. Aero. and Elec. Systems* **32**(2), 554–568.
- Richardson T and Urbanke R 2001 The capacity of low-density parity check codes under message-passing decoding. *IEEE Trans. Info. Theory* **47**, 599–618.
- Rusmevichientong P and Roy BV 2000 An analysis of turbo decoding with Gaussian densities *NIPS 12*, pp. 575–581. MIT Press.

- Sudderth EB, Ihler AT, Freeman WT and Willsky AS 2003 Nonparametric belief propagation *Computer Vision and Pattern Recognition*.
- Tang Z, Pattipati K and Kleinman D 1993 Optimization of detection networks: Part II—Tree structures. *IEEE Trans. Systems, Man and Cybernetics* **23**(1), 211–221.
- Tang ZB, Pattipati KR and Kleinman DL 1991 An algorithm for determining the decision thresholds in a distributed detection problem. *IEEE Trans. Systems, Man and Cybernetics* **21**(1), 231–237.
- Tatikonda S and Jordan MI 2002 Loopy belief propagation and Gibbs measures *Proc. Uncertainty in Artificial Intelligence*, vol. 18, pp. 493–500.
- Thrun S 2006 Affine structure from sound In *Advances in Neural Information Processing Systems 18* (ed. Weiss Y, Schölkopf B and Platt J) MIT Press Cambridge, MA pp. 1355–1362.
- Tsitsiklis JN 1993 Decentralized detection In *Advances in Statistical Signal Processing* (ed. Poor HV and Thomas JB) vol. 2 JAI Press Greenwich, CT pp. 297–344.
- Tsitsiklis JN and Athans M 1985 On the complexity of decentralized decision making and detection problems. *IEEE Trans. Automatic Control* **30**(5), 440–446.
- Van Trees HL 1968 *Detection, Estimation, and Modulation Theory* vol. 1. John Wiley & Sons, New York, NY.
- Varshney PK 1997 *Distributed Detection and Data Fusion*. Springer-Verlag, New York, NY.
- Viswanathan R and Varshney PK 1997 Distributed detection with multiple sensors: Part I—Fundamentals. *Proceedings of the IEEE* **85**(1), 54–63.
- Wainwright MJ 2005 Stable message-passing and convex surrogates: Joint parameter estimation and prediction in coupled Gaussian mixture models *IEEE Workshop on Statistical Signal Processing*.
- Wainwright MJ and Jordan MI 2005 A variational principle for graphical models In *New Directions in Statistical Signal Processing* (ed. et al. SH) MIT Press Cambridge, MA.
- Wainwright MJ, Jaakkola T and Willsky AS 2004 Tree consistency and bounds on the performance of the max-product algorithm and its generalizations. *Statistics and Computing* **14**, 143–166.
- Wainwright MJ, Jaakkola TS and Willsky AS 2003 Tree-based reparameterization analysis of sum-product and its generalizations. *IEEE Trans. Info. Theory* **49**(5), 1120–1146.
- Wainwright MJ, Jaakkola TS and Willsky AS 2005a Exact MAP estimates via agreement on (hyper)trees: Linear programming and message-passing. *IEEE Trans. Info. Theory*.
- Wainwright MJ, Jaakkola TS and Willsky AS 2005b A new class of upper bounds on the log partition function. *IEEE Trans. Info. Theory* **51**(7), 2313–2335.
- Weiss Y 2000 Correctness of local probability propagation in graphical models with loops. *Neural Computation* **12**, 1–41.
- Weiss Y and Freeman WT 2000 Correctness of belief propagation in Gaussian graphical models of arbitrary topology *NIPS 12*, pp. 673–679. MIT Press.
- Welling M 2004 On the choice of regions for generalized belief propagation *Uncertainty in Artificial Intelligence*.
- Whittaker J 1990 *Graphical Models in Applied Multivariate Statistics*. Wiley, New York.
- Wiegerinck W 2005 Approximations with reweighted generalized belief propagation *Workshop on Artificial Intelligence and Statistics*.
- Williams J, Fisher III J and Willsky A 2005 An approximate dynamic programming approach to a communication constrained sensor management problem *Proceedings of 8th International Conference on Information Fusion*.
- Williams JL, Fisher III JW and Willsky AS 2006a Approximate dynamic programming for communication-constrained sensor network management. Technical Report LIDS-TR-2637, Massachusetts Institute of Technology.
- Williams JL, Fisher III JW and Willsky AS 2006b Performance guarantees in dynamic active learning. Technical Report LIDS-TR-2706, Massachusetts Institute of Technology.

- Yedidia J, Freeman WT and Weiss Y 2005 Constructing free energy approximations and generalized belief propagation algorithms. *IEEE Trans. Info. Theory* **51**(7), 2282–2312.
- Zhao F, Shin J and Reich J 2002 Information-driven dynamic sensor collaboration. *IEEE Signal Processing Magazine* **19**(2), 61–72.